Freeform Search

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Term: L13 and collaborative

Display: 50 **Documents in Display Format:** - **Starting with Number** 1

Generate: Hit List [✓ Hit Count] Side by Side Image

Search Clear Interrupt

Search History

DATE: Saturday, March 06, 2004 Printable Copy Create Case

Set Name side by side	**Query**	**Hit Count**	**Set Name** result set
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L14	L13 and collaborative	8	L14
L13	L11 and group	56	L13
L12	L11 and group$	62	L12
L11	L10 and (parent near node)	77	L11
L10	rank$ near node$	259	L10
L9	l5 and rank$	6	L9
L8	L7 and rank$	2	L8
L7	L6 and node$	11	L7
L6	L5 and Hierarch$	19	L6
L5	collaborative near database	30	L5
L4	L3 and network	3	L4
L3	scientific near poster$	10	L3
L2	sciencetific near poster$	0	L2
L1	sciencetif near poster$	0	L1

END OF SEARCH HISTORY

First Hit Fwd Refs

☐ Generate Collection Print

L14: Entry 4 of 8 File: USPT Mar 4, 2003

DOCUMENT-IDENTIFIER: US 6529891 B1
TITLE: Automatic determination of the number of clusters by mixtures of bayesian networks

Brief Summary Text (17):
Although Bayesian networks are quite useful in decision-support systems, Bayesian networks require a significant amount of storage. For example, in the Bayesian network 300 of FIG. 3A, the value of nodes X and Y causally influences the value of node Z. In this example, nodes X, Y, and Z have binary values of either 0 or 1. As such, node Z maintains a set of four probabilities, one probability for each combination of the values of X and Y, and stores these probabilities into a table 320 as shown in FIG. 3B. When performing probabilistic inference, it is the probabilities in table 320 that are accessed. As can be seen from table 320, only the probabilities for Z equaling 0 are stored; the probabilities for Z equaling 1 need not be stored as they are easily derived by subtracting the probability of when Z equals 0 from 1. As the number of parents of a node increases, the table in the node that stores the probabilities becomes multiplicatively large and requires a significant amount of storage. For example, a node having binary values with 10 parents that also have binary values requires a table consisting of 1,024 entries. And, if either the node or one of its parents has more values than a binary variable, the number of probabilities in the table increases multiplicatively. To improve the storage of probabilities in a Bayesian network node, some conventional systems use a tree data structure. A tree data structure is an acyclic, undirected graph where each vertex is connected to each other vertex via a single path. The graph is acyclic in that there is no path that both emanates from a vertex and returns to the same vertex, where each edge in the path is traversed only once. FIG. 3C depicts an example tree data structure 330 that stores into its leaf vertices 336-342 the probabilities shown in table 320 of FIG. 3B. Assuming that a decision-support system performs probabilistic inference with X's value being 0 and Y's value being 1, the following steps occur to access the appropriate probability in the tree data structure 330: First, the root vertex 332, vertex X, is accessed, and its value determines the edge or branch to be traversed. In this example, X's value is 0, so edge 344 is traversed to vertex 334, which is vertex Y. Second, after reaching vertex Y, the value for this vertex determines which edge is traversed to the next vertex. In this example, the value for vertex Y is 1, so edge 346 is traversed to vertex 338, which is a leaf vertex. Finally, after reaching the leaf vertex 338, which stores the probability for Z equaling 0 when X=0 and Y=1, the appropriate probability can be accessed. As compared to a table, a tree is a more efficient way of storing probabilities in a node of a Bayesian network, because it requires less space. However, tree data structures are inflexible in the sense that they can not adequately represent relationships between probabilities. For example, because of the acyclic nature of tree data structures, a tree cannot be used to indicate some types of equality relationships where multiple combinations of the values of the parent vertices have the same probability (i.e., refer to the same leaf vertex). This inflexibility requires that multiple vertices must sometimes store the same probabilities, which is wasteful. It is thus desirable to improve Bayesian networks with tree distributions.

Brief Summary Text (18):
Background Relative to Collaborative Filtering

Brief Summary Text (19):
Collaborative filtering systems have been developed that predict the preferences of a user. The term "collaborative filtering" refers to predicting the preferences of a user based on known attributes of the user, as well as known attributes of other users. For example, a preference of a user may be whether they would like to watch the television show "I Love Lucy" and the attributes of the user may include their age, gender, and income. In addition, the attributes may contain one or more of the user's known preferences, such as their dislike of another television show. A user's preference can be predicted based on the similarity of that user's attributes to other users. For example, if all users over the age of 50 with a known preference happen to like "I Love Lucy" and if that user is also over 50, then that user may be predicted to also like "I Love Lucy" with a high degree of confidence. One conventional collaborative filtering system has been developed that receives a database as input. The database contains attribute-value pairs for a number of users. An attribute is a variable or distinction, such as a user's age, gender or income, for predicting user preferences. A value is an instance of the variable. For example, the attribute age may have a value of 23. Each preference contains a numeric value indicating whether the user likes or dislikes the preference (e.g., 0 for dislike and 1 for like). The data in the database is obtained by collecting attributes of the users and their preferences. It should be noted that conventional collaborative filtering systems can typically only utilize numerical attributes. As such, the values for non-numerical attributes, such as gender, are transposed into a numerical value, which sometimes reduces the accuracy of the system. For example, when a variable has three non-numerical states, such as vanilla, chocolate and strawberry, transposing these states into a numerical value will unintentionally indicate dissimilarity between the states. That is, if vanilla were assigned a value of 1, chocolate 2 and strawberry 3, the difference between each value indicates to the system how similar each state is to each other. Therefore, the system may make predictions based on chocolate being more similar to both vanilla and strawberry than vanilla is similar to strawberry. Such predictions may be based on a misinterpretation of the data and lead to a reduction in the accuracy of the system. In performing collaborative filtering, the conventional system first computes the correlation of attributes between a given user "v" and each other user "u" (except v) in the database. The computation of the "correlation" is a well-known computation in the field of statistics. After computing the correlation, the conventional system computes, for example, the preference of a user "v" for a title of a television show "t" as follows: ##EQU2##

Brief Summary Text (20):
where "pref(t, v)" is the preference of user "v" for title "t," where "<pref(t)>" is the average preference of title "t" by all users, where "pref(t, u)" is the preference of user "u" for title "t," where "corr(u, v)" is the correlation of users "u" and "v," and the sums run over the users "u" that have expressed a preference for title "t." One drawback to this conventional system is that the entire database must be examined when predicting preferences, which requires a significant amount of processing time. One way to improve upon this conventional system is to utilize a clustering algorithm. Using this approach, a collaborative filtering system uses any of a number of well-known clustering algorithms to divide the database into a number of clusters. For example, the algorithms described in KoJain, Algorithms for Clustering Data (1988) can be used. Each cluster contains the data of users whose preferences tend to be similar. As such, when predicting the preferences of one user in a cluster, only the preferences of the other users in the cluster need to be examined and not the preferences of all other users in the database. A collaborative filtering system that utilizes a clustering algorithm receives as input a database, as described above, a guess of the number of clusters and a distance metric. The guess of the number of clusters is provided by an administrator of the collaborative filtering system based on their own knowledge of how many clusters the database can probably be divided into. The distance metric is a metric provided by the administrator for each user in the database that estimates

how similar one user is to each other in the database based on user's preferences and attributes. The distance metric is a range between 0 and 1 with 0 indicating that two users are least similar and 1 indicating that two users are most similar. This similarity is expressed as a numerical value. Each user will have a distance metric for every other user. Thus, the distance metrics are conveniently represented by an N-by-N matrix, where "N" is the number of users. After receiving the number of clusters and the distance metric, the clustering algorithm identifies the clusters. The clustering algorithm outputs a list of the users in the database and a cluster number assigned to each user. To determine the preferences of a user, the other users within that user's cluster are examined. For example, if the system is attempting to determine whether a user would like the television show "I Love Lucy," the other users within that cluster are examined. If there are six other users within the cluster and five out of the six like "I Love Lucy," then it is likely that so will the user. Although utilizing a clustering algorithm may be an improvement over the previously-described conventional system, it has limitations. One such limitation is that the exact number of clusters is determined manually, which renders the algorithm prone to human error. Another limitation is that all attributes are numerical and as such, the values of non-numerical attributes must be transposed into numerical values. Based upon the above-described limitations of conventional collaborative filtering systems, it is desirable to improve collaborative filtering systems.

Drawing Description Text (38):
FIG. 29 depicts a hypothesis-specific Bayesian network in an example relative to collaborative filtering.

Detailed Description Text (12):
A database of observed cases over a set of variables is given. The prediction problem is to learn the statistical relationships among those variables for prediction. The clustering problem is to group the rows of the database into groups so that groups of similar users can be discovered and properties of the groups can be presented. The invention provides a flexible and rich class of models (for both of these problems) and provide algorithms to learn which model from this class of models best fits the data. The class of models employed by the invention is called a mixture of Bayesian networks (MBN). The processes for learning MBNs include several advantageous features including: (a) interleaving parameter and structural search, (b) expected complete model sufficient statistics, and (c) an outer loop for determining the number of states of the discrete hidden variables.

Detailed Description Text (67):
FIG. 18 depicts a diagram of the MBN generator 502 of the exemplary embodiment of FIG. 6. The MBN generator 502 of the exemplary embodiment contains a scoring mechanism 602 and a network adjuster 606. The scoring mechanism 602 receives the expert knowledge 506, the empirical data 504, the test network 608 and a list of nodes 610 as input. After receiving this information, the scoring mechanism 608 generates a score 604 that ranks the nodes of test network 608 as indicated by the list of nodes 610 for goodness. Thus, the score 604 contains a subscore for each node scored. Each subscore indicates how well the portion of the test network involving the node corresponding to the subscore and the parents of the node is at rendering inferences based on the empirical data 504 and the expert knowledge 506. The test network 608 received as input is either the prior network or a test network 608 generated by the network adjuster 606 depending on the circumstances. That is, the scoring mechanism 602 of the exemplary embodiment uses the initial network as the test network for the first invocation of the scoring mechanism. After the first invocation of the scoring mechanism 602, the test network received by the scoring mechanism is the test network 608 generated by the network adjuster. In the exemplary embodiment, a Bayesian network (i.e., the initial network or the test network 608) is stored in memory as a tree data structure where each node in the tree data structure corresponds to a node in the Bayesian network. The arcs of the Bayesian network are implemented as pointers from one node in the tree data

structure to another node. In addition, the probabilities for each node in the Bayesian network are stored in the corresponding node in the tree data structure.

Detailed Description Text (120):
where: "n" is the total number of nodes in the Bayesian network, G.sub.a is the set of leaves for the decision graph in node A of the Bayesian network, r.sub.a is the number of states of node A, and q.sub.a is the number of configurations of the parents of node A, and t.sub.b is the number of configurations of the parents of node a corresponding to b. The term "N.sub.abc " is the expected number of cases where node "a" has a value "c" and the parents of leaf "b" in the decision graph of node "a" are in a state that leads to leaf "b." The term "N.sub.ab " is the sum over "c" of "N.sub.abc." When performing this step, most of the leaves of the decision graph will have the counts already stored from the processing performed in step 1806. However, for those newly generated leaves, created during the processing of step 1808 (discussed below), the counts have not been stored. For these leaves, the Bayesian network generator obtains the counts as described above. After scoring each candidate graph, the Bayesian network generator selects the candidate graph with the best score and stores this graph into the node. Most candidate graphs (other than the first one generated) reflect a single change to a preexisting candidate decision graph where one or more vertices are added. Therefore, when a preexisting decision graph has already been scored, the exemplary embodiment can optimize the scoring step. The exemplary embodiment optimizes the scoring step by obtaining a partial score by only scoring the added vertices, by adding this partial score by the score of the preexisting decision graph, and by subtracting out the portion of the score for parts of the preexisting decision graph that no longer exist (i.e., any vertices or edges that were removed). Those practiced in the art will recognize that a factorable structure prior is required to perform this step.

Detailed Description Text (127):
FIG. 28 depicts a flowchart of the steps performed by the web analyzer 1614 (FIG. 6) of an exemplary embodiment of the present invention. The web analyzer first receives the MBN output by the Bayesian network generator (step 2002). After receiving the MBN network, the web analyzer receives a request from a user containing values (step 2004). In this step, the Bayesian network generator receives observations or values for a number of the nodes of the MBN. For example, the user may input their age and sex. The web analyzer then performs probabilistic inference and ranks the web site categories, business and travel, by the likelihood that the user would like to visit them (step 2006). In this step, any standard Bayesian network inference algorithm, such as the one described in Jensen, Lauritzen, and Olesen, "Bayesian Updating in Recursive Graphical Models by Local Computations", Technical Report R-89-15, Institute of Electronic Systems, Aalborg University, Denmark, may be used by an exemplary embodiment of the present invention. Before using such an inference algorithm, the probabilities of each Bayesian network node is expressed as a table. Such an inference algorithm and its usage is described in greater detail in U.S. patent application Ser. No. 08/602,238, entitled "Collaborative Filtering Utilizing a Belief Network," which has previously been incorporated by reference. If the Bayesian network of an alternative embodiment is used, where the Bayesian network contains cycles, the inference algorithm used is to merely access the decision graph with the values for the nodes received in step 2004 to determine the probability. In this situation, all parent nodes of a node for which inference is requested should have a value provided. After performing probabilistic inference and ranking the nodes reflecting categories of web sites, the web analyzer determines if there are more requests from the user (step 2008). If there are more requests, processing continues to step 2004. However, if there are no more requests, processing ends.

Detailed Description Text (128):
Using a Mixture of Bayesian Networks to Perform Collaborative Filtering

Detailed Description Text (129):
Collaborative filtering has been described in the above-reference application entitled "Collaborative Filtering Utilizing A Belief Network". The mixture of Bayesian networks of the present invention can be employed to carry out the same type of collaborative filtering in a more powerful way. In this case, the collaborative filtering described in the above-referenced application is a special limited case of a collaborative filter using the present invention, a mixture of Bayesian networks. In the special limited case of the prior above-referenced application, there are no arcs in the HSBNs, there are no hidden variables in the HSBNs and there is no structure search step (block 38 of FIG. 12). Thus, the present invention provides a more general and more powerful network for collaborative filtering. Such collaborative filtering is carried out by an appropriate assignment of variables of the mixture of Bayesian networks already described herein. The following is a detailed description of how to assign those variables in order to carry out collaborative filtering using the embodiments of the present invention. FIG. 29 depicts an examplary typical HSBN 2400 within an MBN utilized to determine preferences of a user for a television show. In the exemplary embodiment, Bayesian networks are implemented as an acyclic directed graph data structure with the variables in the Bayesian network corresponding to nodes in the data structure. The Bayesian network 2400 contains a number of variables (or nodes) 2402, 2404, 2406, 2408, 2410, 2412, 2414 and 2416. Two of these variables, 2402 and 2404, reflect causal attributes and are sometimes referred to as causal variables. A "causal attribute" is an attribute that has a causal effect on caused attributes. The caused attributes in the Bayesian network 2400 are reflected by variables 2406, 2408, 2410, 2412 and 2414. These variables are known as caused attributes (or caused variables) because their value is causally influenced by the causal variables. Caused attributes can be of two types: preference or non-preference. Preference caused attributes contain the preferences to be predicted. Non-preference caused attributes are causally influenced by the causal attributes, but are not preferences because the system is not used to predict their value. Non-preference caused attributes are further discussed below. For example, variable 2414 is a preference caused attribute indicating whether a particular user likes the "Power Rangers" television show and variable 2402 is a causal attribute whose value has a causal effect on variable 2414. That is, since "Power Rangers" is primarily enjoyed by children, the younger the age variable, the more likely it is that the user will enjoy the "Power Rangers" show.